Jurnal Ilmiah Teknologi Informasi Asia
Vol.12, No. 2, Tahun 2018
ISSN: 2580-8397 (O); 0852-730X (P)

Implementasi dan Pengujian Modul ESP8266 dengan Aplikasi Android MQTT-Dash pada Jaringan MQTT.

Rizki Priya Pratama
Politeknik Kota Malang
rizkipriyap@gmail.com

ABSTRAK. Penggunaan protokol MQTT pada modul ESP8266 telah banyak dikembangkan. Namun, implementasinya di lapangan, modul ini rawan *hang* akibat ketidakmampuan ESP8266 melakukan koneksi kembalisaat sinyal wifi atau MQTT *broker*terputus. Untuk mengatasi hal itu, metode yang digunakan adalah membuat modul ini melakukan *reset*apabilawaktu yang ditentukan telah habis saat melakukan koneksi kembali. *Timer*pada modul ESP8266 akan segera menghitung jika terjadi gangguan koneksi tersebut selama 10 detik. Jika waktu yang ditentukan habis maka modul ESP8266 akan *restart*. Dari hasil pengujian, dapat disimpulkan bahwa koneksi antara kedelapan modul dengan protokol MQTT dapat berjalan dengan baik.Pesan *publish* dan *subscribe* dapat diterima dan dikirim dengan keberhasilan 100%. Modul-modul tersebut tidak mengalami *hang* meskipun telah berjalan selama 10 hari dengan rata-rata restart sebanyak 268 kali.
Kata Kunci: *ESP8266, MQTT Broker, Protokol MQTT, publish* dan *subscribe*

ABSTRACT.The use of the MQTT protocol in the ESP8266 module has been widely developed. However, its implementation in the field, this module is prone to hang due to the inability of ESP8266 to reconnect when the wifi signal or MQTT broker is disconnected. To overcome this, the method used is to make this module reset if the specified time has expired when reconnecting. The timer on the ESP8266 module will immediately calculate if a connection interruption occurs for 10 seconds. If the specified time is up, then the ESP8266 module will restart. From the test results, it can be concluded that the connection between the eight modules with the MQTT protocol can work well. Publish and subscribe messages can be received and sent with 100% success. These modules do not experience hangs even though they have been running for 10 days with an average restart about 268 times.
Keywords: *ESP8266*,MQTT protocol, *MQTT Broker , publish* and*subscribe*

1. PENDAHULUAN

Seiring perkembangan jaman, teknologi *Internet of Things* (IoT)semakin berkembang dan canggih. Teknologi ini dapat menghubungkan tempat satu ke tempat yang lain dengan mudah, dengan siapa saja dan kapan saja (Zheng,2011). Semua perangkat Iot akan terhubung secara luas melalui jaringan lokal, intranet dan internet. IoT terdiri dari objek, perangkat sensor, infrastruktur komunikasi, unit komputasi dan pemrosesan yang dapat ditempatkan di *cloud*, serta pengambilan keputusan dan sistem pemanggilan tindakan.

Penggunaan teknologi IoT saat ini, sebagian besar data yang ditransmisikan hanya berupa data teks seperti data sensor dan data perintah (Yu, Wang, dan Zhou, Oktober 2010). Penggunaan data-data teks seperti ini banyak dilakukan dengan menggunakan protokol aplikasi ringan MQTT.Protokol *Message Queue Telemetry Transport* (MQTT) adalah protokol yang sangat sederhana dan ringan (Sankar, September 2016). Keunggulan protokol MQTT adalah arsitekturnya terdiri dari *publish/subscribe* sehingga protokolnya ringan, penggunaan *broker* yang dapat di *password* sehingga menjamin keamaan data, serta mempunyai Qos yang bervariasi bergantung pada kondisi jaringan. Protokol MQTT ini juga dapat digunakan secara multicast dan mendukung SSL (Stanford, November 2013).

Banyak publikasi yang pernah dilakukan untuk aplikasi protokol MQTT. Pada artikel (Amrutkar, 2016), protokol MQTT digunakan untuk *home automation* melalui jaringan internet dan GSM.Artikel (Patel, 2015) membahas protokol MQTT digunakan untuk *home automation* dengan koneksi jaringan nirkabel. Sedangkan artikel (Pratama, 2017).,aplikasi Node-Red pada raspberry piuntuk membaca sensor tegangan dan arus listrik rumah melalui jaringan MQTT. Semua artikel tersebut menggunakan modul ESP8266 sebagai *client* MQTT dan belum membahas masalah kehandalan pada perangkat ini.

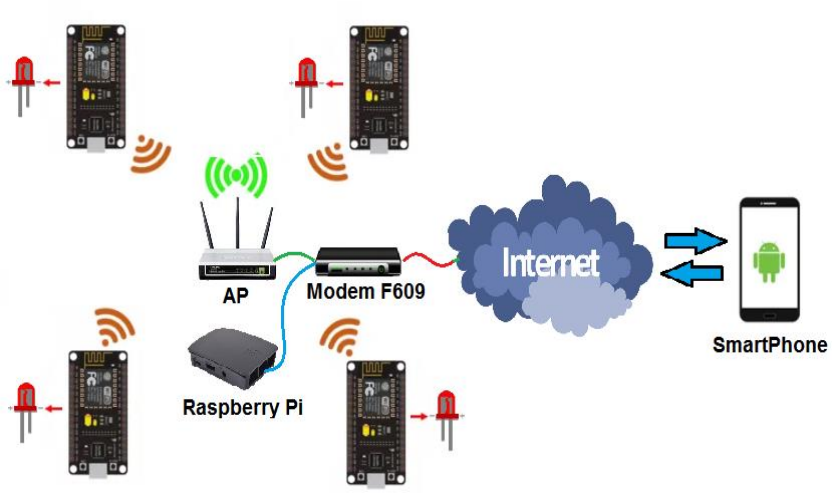
Masalah yang sering timbul pada modul ESP8266 adalah sinyal WiFi yang diterima sering hilang atau mengecil. Modul ini akan berusaha untuk melakukan koneksi kembali ke jaringan tersebut. Namun jika tidak berhasil untuk berkoneksi kembali, maka modul akan *hang* dan perlu untuk menekan tombol *reset* untuk bekerja kembali.

Artikel ini akan membahas bagaimana mengimplementasikan perangkat ESP8266 pada protokol MQTT sehingga perangkat ini stabil dan terkoneksi MQTT broker tanpa mengalami *hang*/*crash*. Metode yang ditawarkan adalah mereset / merestart modul ESP8266 dengan sejumlah timer, sebelum modul ini mengalami *hang* / *crash*.Ujicobanya dilakukan dengan delapan buah perangkat ESP8266yang dikendalikan menggunakan aplikasi android MQTT-Dashuntuk menghidupkan dan mematikan lampu LED. Kedelapan buah ESP8266 ini diprogram dengan topik yang berbeda-beda sehingga dapat dikenali oleh jaringan MQTT.

2. MetodePenelitian

2.1. Konsep umum

Secara keseluruhan blok diagram sistem dapat diamati pada Gambar 1. Prinsip kerja dari sistem ini adalah *smart phone* android melalui aplikasi MQTT-Dash mengendalikan LED, dimana setiap LED mulai dari satu hingga delapan dihubungkan masing-masing ke modul ESP8266. Akses poin (AP) sekaligus modem berfungsi untuk menghubungkan beberapa modul ESP8266 dengan jaringan internet. Pemilihan jenis AP ini berperan penting dalam hal ketahanan dan kestabilan sistem komunikasi WIFI. Raspberry pisebagai MQTT *broker*, digunakan sebagai *server* protokol MQTT untuk meneruskan dan memfilter data dan topik. Sistem dapat diakses dari luar jaringan lokal melalui modem dengan konfigurasi DDNS sehingga *smart phone* dapat mengendalikan masing-masing lampu baik dari dari jaringan lokal maupun dari internet.



Gambar 1. Blok diagram sistem secara keseluruhan

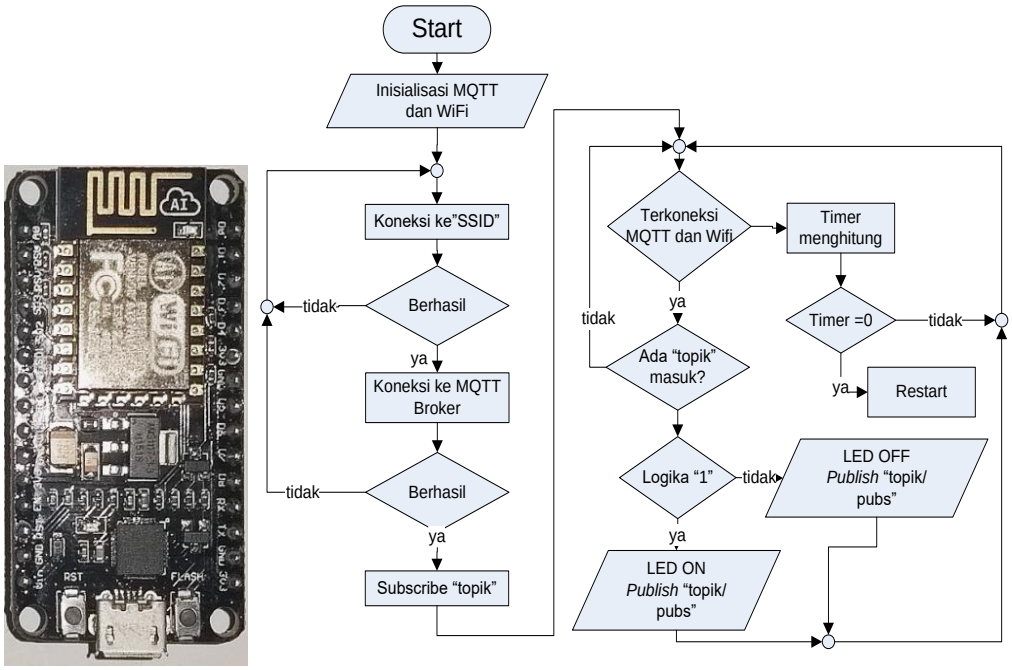
2.2 Instalasi MQTT *broker* Raspberry PI

Raspberry pi 3 adalah komputer berukuran sebesar kotak kecil dengan ukuran 9,5 x 7,5 x 3 cm. Raspberry pi 3 ini memiliki sistem Broadcom BCM2837 SoC yang terdiri dari ARM Cortex-A53 processor 1.2 GHz 64-bit, VideoCore IV GPU, dan dilengkapi dengan RAM sebesar 1GB, Bluetooth dan wireless lan.

Raspberry pi yang digunakan ini, bekerja dengan OS raspbian sebagai *server* untuk memfasilitasi jaringan MQTT. MQTT*broker* yang digunakan adalah MQTT Mosquitto yang telah banyak digunakan.MQTT *broker* pada raspberry pi ini dapat diatur agar *server* tidak dapat diakses oleh klien yang tidak mempunyai *username* dan *password* yang benar dan *port* yang digunakan adalah 1880.

2.3. PemrogramanESP8266 dengan protokol MQTT

Modul yang digunakan pada artikel ini, adalah modul ESP8266-12E. Modul tersebut dapat dilihat pada Gambar 2. Modulini terbentuk dari ESP8266-12E dan USBtoTLLsehingga modul tersebut dapat langsung diprogram tanpa perlu perangkat *downloader* tambahan. Selain itu, modul ini disertai dengan terminal *input-output* (IO) untuk antarmuka komponen-komponen elektronik.



Gambar 2. Modul ESP8266 dan Diagram alir program ESP8266

ESP8266 diprogram menjadi klien MQTT untuk mengendalikan LED dan menerima perintah dari aplikasi MQTT-Dash. Perancangan perangkat lunak ini membutuhkan paket *library*ESP8266WiFi, PubSubClient, dan ESP8266WebServer.*Library*ESP8266WiFidigunakan untuk mengaktifkan Wifi baik sebagai mode akses point maupun mode *station*. *Library* PubSub Client digunakan mengkonfigurasi protokol MQTT, sedangkan *library* ESP8266 WebServer digunakan untuk melayani proses penampilan *web* seperti menyimpan *syntax / script* dari *web* pengaturan SSID. Mode *station* digunakan agar modul ini terhubung ke akses point, dengan menggunakan perintah WiFi.mode(WIFI_STA) dan dapat terhubung dengan jaringan WIFI dengan menggunakan perintah WiFi.begin(ssid,password). Perintah WiFi.status () digunakan untuk memastikan bahwa modul telah terhubung pada akses poin.

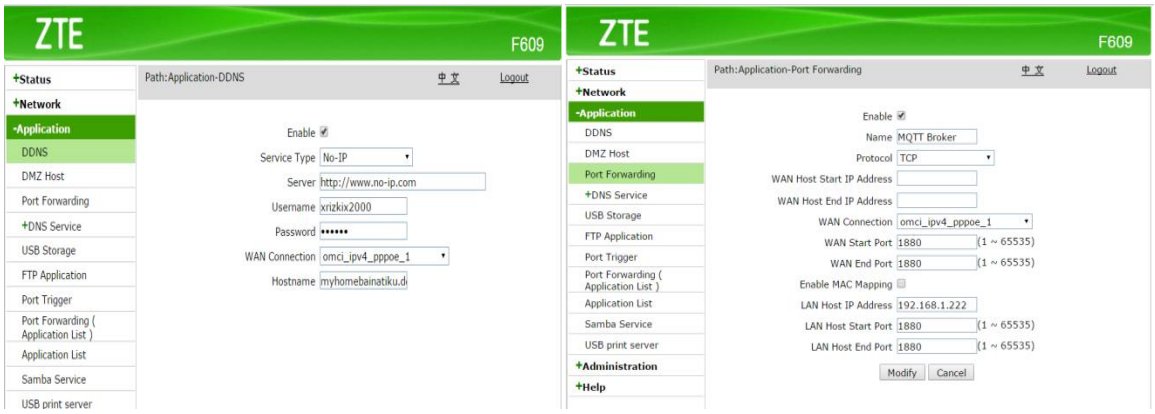
Untuk melayani protokol MQTT, ESP8266 mengirimkandata *subscribe* sesuai dengan topik "/test/lampu01" dengan perintah PubSubClient.subscribe (test/lampu01,1) dengan angka '1' terakhir merupakan *Quality of Service* (QoS). Sedangkan modul-modul ESP8266 yang lainnya, topik yang digunakan adalah "/test/lampu02" hingga "/test/lampu08" sesuai dengan urutan kesatu hingga kedelapan. Selain itu, ESP8266 ini dilengkapi dengan data *publish* dengan topik "/status", yang mana topik ini akan di *publish* secara berkala ketika kondisi ESP8266 dalam keadaan aktif. Data topik ini berisikan informasi-informasi tegangan catu dan jumlah kondisi *restart*, dan juga ditambahkan informasilama waktu / durasi koneksi dengan akses poin dan MQTT *broker* dari keadaan ESP826 mulai hidup.Topik selanjutnya yang akan di-*publish* oleh ESP8266 adalah topik "/esp/lampu01/pubs". Topik ini akan *publish* jika ada perintah untuk mematikan dan menghidupkan LED pada topik "/test/lampu01". Data topik "/esp/lampu01/pubs" ini akan memberikan informasi bahwa data LED dalam keadaan "0" atau "1".

Gambar 2. merupakan diagram alir program ESP8266. Proses pengecekan koneksi WiFi atau koneksi MQTT *broker* terletak pada program *loop.* Proses ini akan menentukan kondisi ESP8266 mengalami keadaan reset atau tidak. Jika koneksi WiFi atau koneksi MQTT terputus, maka program akan menunggu koneksi kembali sekaligus timer menghitung selama 10 detik. Jika koneksi keduanya tidak terjadi dan waktu telah habis maka ESP8266 akan *restart.*

2.4. Konfigurasi Modem dan DDNS .

MQTT *broker* yang digunakan pada artikel ini adalah MQTT *broker* dari rapsberry pi. MQTT *broker* dari rapsberry pi perlu dilakukan pengaturan. MQTT *broker*dari raspberry pi ini dapat diakses dari internet melalui IP publik yang didapatkan dari layanan internet indihome. Namun, IP ini merupakan IPyang sering berubah-ubah (dinamik). Dengan adanya program dari pihak ketiga, perubahan IP ini akan selalu disimpan secara otomatis sehingga *domain* yang didaftarkan dapat selalu mengenali IP tersebut. *Domain* yang

didapatkan dari website : no-ip.com bernama "myhomebainatiku.ddns.net". Nama *domain* ini dimasukkan dalam pengaturan di modem ZTE F609 seperti pada gambar 3.

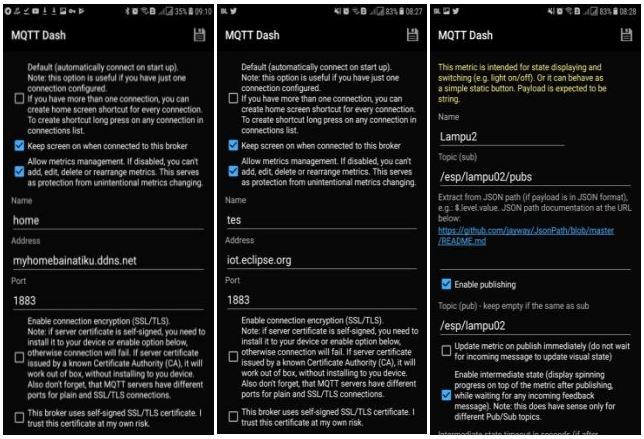


Gambar 3. Pengaturan DDNS dan *Port Forwarding*

Raspberry pi yang sudah di-*install* MQTT broker mempunyai alamat 192.168.1.222:1880. Alamat tersebut dimasukan ke modem pada menu *Port Forwarding*, sehingga domain yang bernama "myhomebainatiku.ddns.net pada port 1880 bertugas untuk melaksanakan fungsi MQTT *broker*.

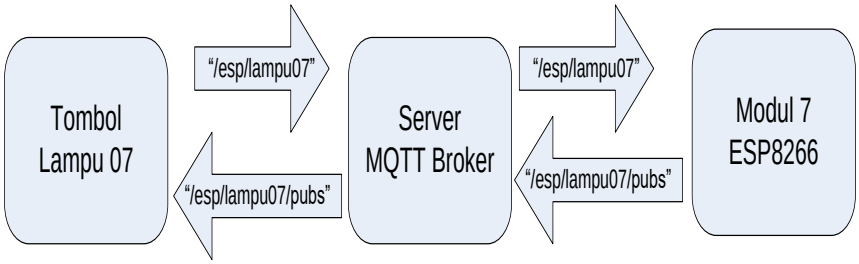
2.4. Instalasi MQTT-Dash android.

Aplikasi MQTT Dash ini membuat *smart phone* menjadi klien MQTT untuk mengontrol lampu LED.*Server*MQTT *broker* dimasukkan pada pengaturan untuk menjalankan aplikasi ini. *Server*MQTT *broker*raspberry pimempunyai alamat bernama : myhomebainatiku.ddns.net dengan port 1880. Nama server ini akan dimasukkan pada konfigurasi pada aplikasi MQTT-Dash android seperti pada Gambar 4.



Gambar 4. Pengatuan Alamat MQTT Broker dan topik pada aplikasi MQTT-Dash android.

Aplikasi ini digunakan mengirimkan data *publish* dan menerima data *subscribe* secara bergantian dengan topik yang berbeda. Sebagai contoh, saat ada penekanan tombol lampu7 pada Aplikasi MQTT-Dash, maka *smartphone* akan mengirimkan data publish "1" dengan topik "/esp/lampu07", untuk menghidupkan lampu LED.Kemudian setelah modul ESP ke 7 menerima topik tersebut, maka led hidup dan ESP akan mengirimkan dapat publish "1" dengan topik "/esp/lampu07/pubs" dan diterima oleh Lampu7 pada MQTT-Dash yang mempunyai data subscribe "/esp/lampu07/pubs" sehingga tombol Lampu7 akan berubah menjadi kondisi *check.* Penjelasan diatas dapat dilihat padaGambar 5.

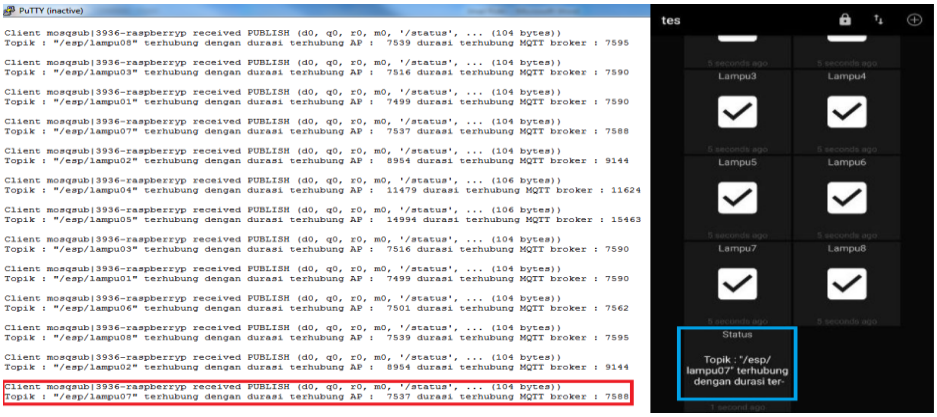


Gambar 5. Pengatuan Alamat MQTT Broker pada aplikasi MQTT-Dash android.

3. Hasil dan Pembahasan

3.1. Pengujian Tahap *Publish* Data oleh ESP8266 ke*Broker*

Pengujian ini bertujuan untuk mengetahui keberhasilan masing-masing ESP8266 untuk mengirimkan pesan *publish* ke *broker* dan *broker* meneruskan pesan tersebut ke aplikasi MQTT-Dash. ESP8266 mem-*publish* topik "/status" dengan data kalimat "*Topik :"/esp/lampu7" terhubung dengan durasi terhubung AP : 8474 durasi terhubung MQTT broker : 8521*", dimana semua ESP8266 melakukan hal yang sama. Namun, informasi topik lampu dan nilai-nilai waktu terhubung yang berbeda-beda. Hasil pengujian *publish* data dari ESP8266 ke MQTT *broker* dapat dilihat pada Gambar 6a sedangkan Gambar 6b merupakan hasil pengujian *broker* meneruskan pesan ke aplikasi MQTT-Dash.



a b

Gambar 6. Hasil pengujian Tahap *Publish* Data oleh ESP8266 ke Broker

Tabel hasil percobaan pada pengiriman data dari ESP8266 sampai ke MQTT *broker*, kesesuaian data yang dikirim dan diterima dapat disajikan dalam Tabel 1.

Tabel 1. Contoh Tabel dan Keterangannya

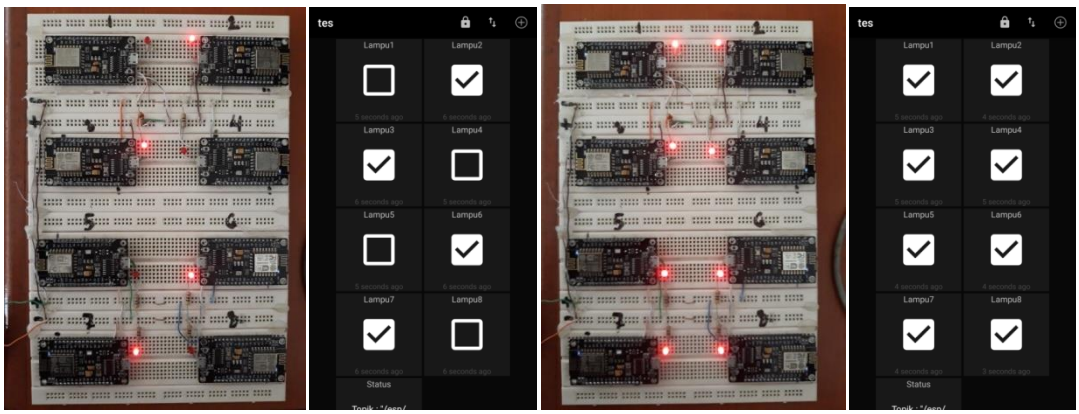
Percobaan ke -	Data yang diterima	Data sesuai
1	Topik : "/esp/lampu01" terhubung dengan durasi terhubung AP : 11564 durasi terhubung MQTT broker : 12224	Ya
2	Topik : "/esp/lampu02" terhubung dengan durasi terhubung AP : 8981 durasi terhubung MQTT broker : 23734	Ya
3	Topik : "/esp/lampu03" terhubung dengan durasi terhubung AP : 7984 durasi terhubung MQTT broker : 24710	Ya
4	Topik : "/esp/lampu04" terhubung dengan durasi terhubung AP : 7977 durasi terhubung MQTT broker : 24607	Ya
5	Topik : "/esp/lampu05" terhubung dengan durasi terhubung AP : 7986 durasi terhubung MQTT broker : 24611	Ya
6	Topik : "/esp/lampu06" terhubung dengan durasi terhubung AP : 10511 durasi terhubung MQTT broker : 11295	Ya

7	Topik : "/esp/lampu07" terhubung dengan durasi terhubung AP : 5540 durasi terhubung MQTT broker : 38297	Ya
8	Topik : "/esp/lampu08" terhubung dengan durasi terhubung AP : 5035 durasi terhubung MQTT broker : 35829	Ya

Dari data hasil pengamatan yang dilakukan terlihat bahwa setiap ESP8266 mampu untuk mem-*publish* datanya tanpa ada kerusakan data dan dengan persentasi keberhasilan 100%.

3.2. Pengujian Tahap *Subscribe* Data dari MQTT-Dash ke ESP8266

Pengujian ini bertujuan untuk mengetahui persentase keberhasilan ESP8266 dalam menerima data dari aplikasi MQTT-dash melalui *broker*. Hasil pengujian didapatkan dari setiap percobaan penekanan tombol pada aplikasi MQTT-Dash yang akan mempengaruhi kondisi LED pada ESP8266. Modul ESP8266 disusun di atas papan percobaan / *project board* dari urutan kesatu hingga kedelapan seperti pada Gambar 6. Gambar 6 menunjukkan bahwa ketika lampu pada aplikasi MQTT-Dash ditekan maka lampu LED pada papan percobaan juga hidup.

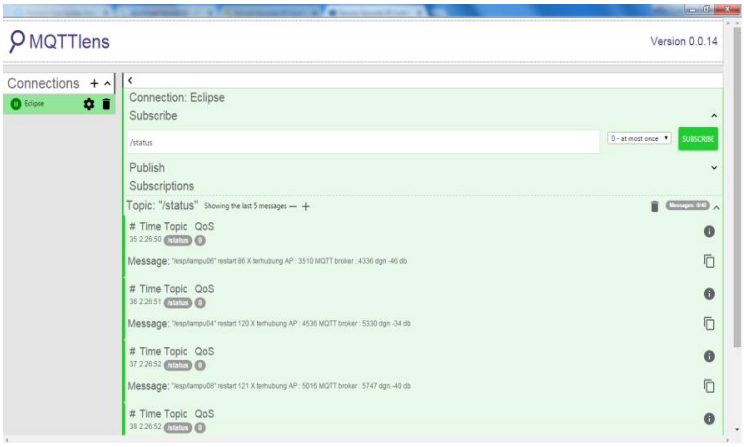


Gambar 7. Hasil pengujian *Subscribe* data dari MQTT-Dash ke ESP8266

Hasil pengamatan dari percobaan menggunakan kedelapan ESP8266 di papan percobaan dengan dikendalikan aplikasi MQTT-Dash dapat disimpulkan bahwa persentase seluruh lampu LED dapat dikendalikan dari MQTT-Dash adalah 100%.

3.3. Pengujian kehandalan modul ESP8266

Pengujian ini bertujuan untuk melihat kehandalan modul ESP8266 dan mengetahui jumlah *restart* ESP8266 dalam memelihara koneksi ke MQTT *broker* dan Wifi.Pengujian ini dilakukan dengan menghidupkan perangkat ESP8266 selama beberapa hari dengan mengamati ESP8266 dalam kondisi berjalan atau *hang*. Hasil pengujian ini dipantau melalui aplikasi *add-ons* google chrome yaitu MQTT Lens. Gambar 7 merupakan aplikasi MQTT Lens yang berisi informasi jumlah *restart* ESP8266 yang bekerja selama 3 hari.



Gambar 8. Tampilan MQTT Lens berisi informasi jumlah *restart* pada ESP8266.

Tabel 2. Kondisi perangkat ESP8266 setelah 24 jam

	ESP-1	ESP-2	ESP-3	ESP-4	ESP-5	ESP-6	ESP-7	ESP-8
Jumlah restart (x)	18	18	14	17	15	9	20	17
Kondisi	Jalan	Jalan	Jalan	Jalan	Jalan	Jalan	Jalan	Jalan

Tabel 3. Kondisi perangkat ESP8266 setelah 10 hari

	ESP-1	ESP-2	ESP-3	ESP-4	ESP-5	ESP-6	ESP-7	ESP-8
Jumlah restart (x)	269	239	255	286	284	247	267	299
Kondisi	Jalan	Jalan	Jalan	Jalan	Jalan	Jalan	Jalan	Jalan

Hasil pengamatan menunjukkan bahwa semua ESP8266 mampu bekerja selama 10 hari dan tidak mengalami *hang* dengan kondisi restart rata-rata sebanyak 268 kali.

4. Kesimpulan

Beberapa percobaan yang telah dilakukan diatas dapat disimpulkan bahwa sistem dengan protokol MQTT dapat berjalan dengan baik. Pesan *publish* dan *subscribe* dapat diterima dan dikirim dengan keberhasian 100%, menggunakan *broker*raspberry pi. Dengan metode yang telah diterapkan, ESP8266 tidak mengalami *hang* meskipun telah berjalan selama 10 hari dengan rata-rata restart sebanyak 268 kali.

DAFTAR RUJUKAN

Amrutkar, R., Vikharankar S., Ahire, L. (2016). Security : Smart Homes Using Internet of Things (IOT).*International Engineering Research Journal (IERJ)*, 2(2), 558-561.

Patel K.K, Patoliya J, Patel H. (2015). Low Cost Home Automation with ESP8266 and,Lightweight protocol MQTT.*Transactions on Engineering and Sciences*, 3(6),14-19

Pratama, R. P.,(2017). Aplikasi Wireless Sensot ESP8266 untuk Smart Home, 1–10.

Hartalkar, T., Bhore. S.,, Borawake, K. (2015) GSM based Home Automation using MQTT. *International Journal of Engineering Technology, Management and Applied Sciences* (9):93-98.

Sankar, P. (September 2016). A Secure and Fast Authentication implementation between the Entities using Trust Aware Algorithm.*International Innovative Research Journal of Engineering and Technology***,** 2 (1).

Stanford-Clark, A., Truong,HL.,. (November 2013). MQTT For Sensor Networks (MQTT-SN) Protocol Specification Version 1.2.

Yu, Y.,Wang,J., and Zhou G. (Oktober 2010). The Exploration in the Education of Professionals in Applied Internet of Things Engineering.*4th International Conference on Distance Learning and Education (ICDLE).*

Zheng,J., Simplot-Ryl,D., Bisdikian, C., and Mouftah, H. (2011). The Internet of Things. *IEEE Communications Magazine*, 49(11), 30-31.